NASA/CR-2000-210319
ICASE Report No. 2000-29

An Efficient Parallel Multigrid Solver for 3-D Convection-dominated Problems

Ignacio M. Llorente and Manuel Prieto-Matías
Universidad Complutense, Madrid, Spain

Boris Diskin
ICASE, Hampton, Virginia

Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA

Operated by Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NAS1-97046

August 2000

AN EFFICIENT PARALLEL MULTIGRID SOLVER FOR 3-D CONVECTION-DOMINATED PROBLEMS*

IGNACIO M. LLORENTE†, MANUEL PRIETO-MATÍAS‡, AND BORIS DISKIN§

Abstract. Multigrid algorithms are known to be highly efficient in solving systems of elliptic equations. However, standard multigrid algorithms fail to achieve optimal grid-independent convergence rates in solving non-elliptic problems. In many practical cases, the non-elliptic part of a problem is represented by the convection operator. Downstream marching, when it is viable, is the simplest and most efficient way to solve this operator. However, in a parallel setting, the sequential nature of marching degrades the efficiency of the algorithm. The aim of this report is to present, evaluate and analyze an alternative highly parallel multigrid method for 3-D convection-dominated problems. This method employs semicoarsening, a four-color plane-implicit smoother, and discretization rules allowing the same cross-characteristic interactions on all the grids involved to be maintained. The resulting multigrid solver exhibits a fast grid-independent convergence rate for solving the convection-diffusion operator on cell-centered grids with stretching. The load imbalance below the critical level is the main source of inefficiency in its parallel implementation. A hybrid smoother that degrades the convergence properties of the method but improves its granularity has been found to be the best choice in a parallel setting. The numerical and parallel properties of the multigrid algorithm with the four-color and hybrid smoothers are studied on SGI Origin 2000 and Cray T3E systems.

Key words. robust multigrid methods, convection dominated problems, parallel multigrid methods

Subject classification. Applied Mathematics

1. Introduction. The convergence properties of multigrid algorithms are defined by two factors: (1) the smoothing rate, which describes the reduction of high-frequency error components, and (2) the quality of the coarse-grid correction, which is responsible for the dumping of smooth error components. In elliptic problems, all the smooth fine-grid components are well approximated on the coarse grid built by standard (full) coarsening. In non-elliptic problems, however, some fine-grid *characteristic* components that are much smoother in the characteristic direction than in other directions, cannot be approximated with standard multigrid methods (see [2, 3, 4, 7]).

Several approaches aimed at curing the characteristic-component problem have been studied in literature. These approaches fall into two categories: (1) development of a suitable relaxation scheme to eliminate not only high-frequency error components but the characteristic error components as well; (2) devising an adjusted coarse-grid operator to approximate well the fine-grid characteristic error components.

In many practical problems appearing in computational fluid dynamics (CFD), the non-elliptic part is

*This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046 while the authors were in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681-2199. Ignacio M. Llorente and Manuel Prieto-Matías were also supported in part by the Spanish research grant CICYT TIC 99/0474.

†Departamento de Arquitectura de Computadores y Automática, Universidad Complutense, 28040 Madrid, Spain (email: llorente@dacya.ucm.es)

‡Departamento de Arquitectura de Computadores y Automática, Universidad Complutense, 28040 Madrid, Spain (email: mpmatias@dacya.ucm.es)

§Institute for Computer Applications in Science and Engineering, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23681-2199 (email: bdiskin@icase.edu)

represented by convection. For convection, the most efficient and comprehensive relaxation is downstream marching. If the target discretization is a stable upwind scheme, the downstream marching reduces all (high-frequency and smooth) error components, *solving* a non-linear convection equation in just a few sweeps (a single downstream sweep provides the exact solution to a linearized problem). Incomplete LU (ILU) decomposition methods act similarly, given a suitable ordering [6]. The downstream marching technique was successfully applied in solving many CFD problems associated with non-recirculating flows (see, e.g., [4]). However, if the discretization is not fully upwind (e.g., upwind biased) the downstream marching in its pure form is not viable. One of the most efficient (also marching type) alternatives often applied to the schemes that cannot be directly marched is a defect-correction method (see, e.g., [29, 41, 42]). Usually the efficiency of these methods is quite satisfactory. Sometimes, however, the convergence rate of the defect-correction method is grid dependent (see [8, 9]). Another, very important, drawback associated with all marching and ILU methods is a low parallel efficiency, because the efficiency of these methods is essentially based on the correctness of the sequential marching order.

In methods belonging to the second category, most of the operations can be performed in parallel. These methods are much more attractive for massive parallel computing. The necessary requirements for coarse-grid operators used in second-category methods were formulated in [46]. Among the options available in conjunction with full coarsening are Galerkin coarsening [47], matrix-dependent operators [6], and corrected coarse-grid operators [17]. Analysis in [46] showed that all these methods have certain drawbacks.

Another way to construct an appropriate coarse-grid operator is to employ semicoarsening [7]. The multigrid method evaluated in this report uses semicoarsening together with a well-balanced correction of discrete operators to maintain the same cross-characteristic interaction (CCI) on all the grids. The relaxation scheme employed in this algorithm is a four-color plane-implicit scheme enabling a very efficient parallel implementation. The resulting algorithm is an efficient highly parallel method for solving the three-dimensional (3-D) convection operator defined on cell-centered grids with stretching.

When studying the optimal parallel implementation of a numerical algorithm, one should consider both the numerical and parallel properties. The best approach in a sequential setting may not be the optimal one on a parallel computer. The multigrid method proposed in this report exhibits a very fast convergence rate but the granularity of its smoother (four-color relaxation) is finer than that of other common smoothers as zebra or damped Jacobi. In order to improve the granularity of the solver, we have studied a hybrid smoother that uses a four-color, zebra or damped Jacobi update depending on the level. As we will show, although this smoother degrades the convergence properties of the original method, it improves the execution time of the multigrid cycle in a parallel setting, and so becomes a trade-off between numerical and architectural properties.

Section 2 formulates the model problem for the convection equation and introduces the notion of the low-dimensional prototype. The multigrid method for the one-dimensional prototype is described in Section 3. The 3-D discretizations and the difficulties encountered in multigrid methods solving the convection operator are explained in Section 4. Section 5 presents the multigrid cycle for the full-dimensional problem. Section 6 includes numerical results confirming the efficient solution of the convection equation on uniform and stretched grids. Section 7 demonstrates an extension of the tested method to the convection-diffusion equation. The parallel properties of the MPI implementation of the code on SGI Origin 2000 and Cray T3E systems are discussed in Section 8. Finally, the main conclusions and future research directions are presented in Section 9.

2. Convection Equation. The model problem studied in this Section is the 3-D constant-coefficient convection equation

$$\frac{1}{|\bar{a}|}\Big(\bar{a}\cdot\nabla\Big)U = f(x,y,z), \tag{2.1}$$

where $\bar{a}=(a_1,a_2,a_3)$ is a given vector and $|\bar{a}|=\sqrt{a_1^2+a_2^2+a_3^2}$. The solution $U(x,y,z)$ is a differentiable function defined on the unit square $(x,y,z)\in[0,1]\times[0,1]\times[0,1]$. Let $t_y=a_2/a_1$ and $t_z=a_3/a_1$ be non-alignment parameters. For simplicity, we assume $a_1\geq a_2,a_3\geq 0$ and, therefore, $1\geq t_y,t_z\geq 0$.

Equation (2.1) can be rewritten as

$$\partial_\xi U = f(x,y,z), \tag{2.2}$$

where $\xi=\frac{x+t_y y+t_z z}{\sqrt{1+t_y^2+t_z^2}}$ is a variable along the characteristic of (2.1). Equation (2.1) is subject to Dirichlet boundary conditions at the inflow boundary $x=0$ and periodic conditions in the y and z directions

$$U(0,y,z)=g(y,z), U(x,y,z)=U(x,y+1,z), U(x,y,z)=U(x,y,z+1), \tag{2.3}$$

where $g(y,z)$ is a given function.

In the 3-D constant-coefficient case, which is studied in this paper, characteristics of (2.1) are straight lines (*characteristic lines*) aligned with the velocity direction. A function is called *a characteristic component* if it is much smoother in the characteristic direction than in other directions.

The problem (2.2) is discretized on the 3-D Cartesian uniform grid with mesh sizes h in the three directions (target grid). Let u_{i_1,i_2,i_3} be a discrete approximation to the solution $U(x,y,z)$ at the point $(x,y)=(i_1h,i_2h,i_3h)$. To derive a proper discretization, we exploit the idea of a *low-dimensional prototype* introduced in [3]. Briefly, the low-dimensional prototype is a convenient discretization of the differential operator in the grid induced on the characteristic manifold by the intersections of this manifold with the full-dimensional Cartesian grid. For our studies, we choose the low-dimensional prototype to be the (one-dimensional) first-order accurate discretization of the first derivative, corresponding to the pure upwind scheme with an additional streamwise dissipation

$$\frac{1}{h_\xi}\Big(u_{i_1,i_2,i_3}-u_{i_1-1,i_2-t_y,i_3-t_z}\Big)-\frac{\lambda}{h_\xi}\Big(u_{i_1+1,i_2+t_y,i_3+t_z}-2u_{i_1,i_2,i_3}+u_{i_1-1,i_2-t_y,i_3-t_z}\Big)=f_{i_1,i_2,i_3}, \tag{2.4}$$

where the discretization of the right-hand-side function is $f_{i_1,i_2,i_3}=f(i_1h,i_2h,i_3h)$ and $h_\xi=h\sqrt{1+t_y^2+t_z^2}$.

3. Multigrid Cycle for Low-Dimensional Prototype. In this section, we define a multigrid cycle for the low-dimensional prototype problem. All the components of this cycle serve as bases for constructing corresponding components of the 3-D multigrid cycle described in Sections 4 and 5.

On the grid induced on the characteristic line, the one-dimensional prototype can be reformulated as

$$L_\lambda u_i \equiv \frac{1}{h_\xi}\begin{cases} 2\Big(u_i-u_{i-1}\Big)-\lambda\Big(u_{i+1}-3u_i+2u_{i-1}\Big), & i=1,\\ \Big(u_i-u_{i-1}\Big)-\lambda\Big(u_{i+1}-2u_i+u_{i-1}\Big), & i=2,3,\dots N-1,\\ \Big(u_i-u_{i-1}\Big)-\lambda\Big(2u_{i+1}-3u_i+u_{i-1}\Big), & i=N,\\ 2\Big(u_i-u_{i-1}\Big), & i=N+1. \end{cases} \tag{3.1}$$

See Figure 1 for the pictorial explanation of the discretization grids.

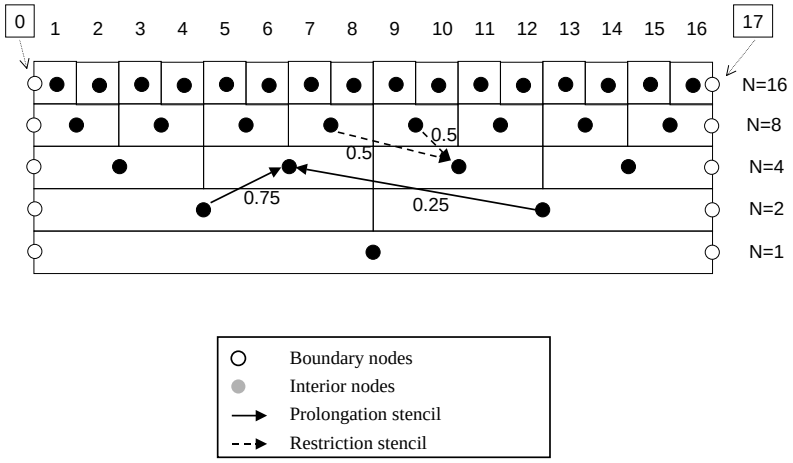

FIG. 1. *Cell-centered discretization of the low-dimensional prototype.*

3.1. Relaxation Scheme. A multicolor (with p colors) relaxation order is defined as follows. One multicolor relaxation iteration consists of p sweeps, each one passes through (approximately) N/p cells. In the first sweep, all the points with coordinates $i = 1 + jp$ (j is a non-negative integer) are relaxed; in the second, all the points with coordinates $i = 2 + jp$ are relaxed (in this sweep the new values at previously relaxed points are used); and so on until all the points are updated.

The efficiency of a multicolor relaxation scheme applied to the convection operator improves when more colors (in the streamwise direction) are used. In fact, the downstream relaxation is an extreme case where the number of colors coincides with the number of grid nodes in the streamwise direction. However, the main subject of this paper is parallel multigrid algorithms, therefore, relaxation schemes with only a few colors are considered. Such schemes are very efficient in parallel implementation. In our tests, we have experimented with $p = 4$ because it appears to be a good tradeoff between parallel and convergence properties. In fact, as we will show in Section 8, a suitable combination of different multicolor smoothers in different grids, where the four-color smoother is always applied for fine grids above some critical level, is found to be the optimal solution in a parallel setting.

The value of $\lambda = 0.25$ has been chosen to provide a good smoothing factor for the four-color relaxation scheme.

3.2. Intergrid Transfers. The cell-centered location of the unknowns suggests that the coarse-grid nodes are shifted relative to those of the fine-grid. In our method, we add two additional nodes to all the grids. These nodes are located precisely at the inflow and outflow boundaries (see Figure 1).

The first type of intergrid transfers encountered in multigrid methods is the computation of a coarse-grid approximation to the current fine-grid residual function. In the proposed method, an upwind restriction is

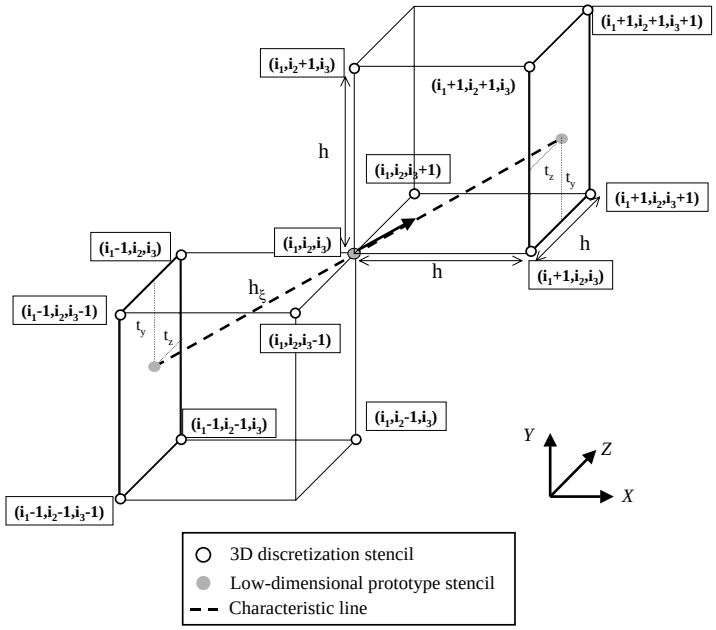

FIG. 2. *Uniform-grid discretization stencil.*

used

$$\left\{ \begin{array}{lcl} R_1 & = & \frac{1}{2} r_1, \\ R_i & = & \frac{1}{2}\Big(r_{2i-1} + r_{2i-2} \Big), \end{array} \right. \tag{3.2}$$

where R and r denote the coarse- and fine-grid residual functions respectively (see Figure 1).

The second type of intergrid transfers is the coarse-grid correction to the current fine-grid solution. The coarse-grid correction V is prolongated (linear interpolation) to the fine grid by

$$\left\{ \begin{array}{lcl} v_1 & = & \frac{1}{2} V_1, \\ v_{2i} & = & \frac{1}{4}\Big(V_{i+1} + 3V_i \Big), \\ v_{2i+1} & = & \frac{1}{4}\Big(3V_{i+1} + V_i \Big), \end{array} \right. \tag{3.3}$$

where v is the correction to the fine-grid solution approximation (see Figure 1).

The idea of using a first-order upwind restriction operator and a linear second-order prolongation operator was borrowed from [17].

4. Full-Dimensional Discretizations.

4.1. Uniform-Grid Discretization. The 3-D discretization is obtained from the low-dimensional prototype discretization (2.4) by replacing function values at the ghost points (points with fractional indexes) by weighted averages of the values at adjacent genuine grid points. The resulting *narrow* discretization scheme

is defined by

$$
\begin{aligned}
L^h u_{i_1,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (1+2\lambda)u_{i_1,i_2,i_3} \\
& -(1+\lambda)\Big((1-t_z)\,\big((1-t_y)u_{i_1-1,i_2,i_3}+t_y u_{i_1-1,i_2-1,i_3}\big) \\
& \qquad +t_z\,\big((1-t_y)u_{i_1-1,i_2,i_3-1}+t_y u_{i_1-1,i_2-1,i_3-1}\big)\Big) \\
& -\lambda\Big((1-t_z)\,\big((1-t_y)u_{i_1+1,i_2,i_3}+t_y u_{i_1+1,i_2+1,i_3}\big) \\
& \qquad +t_z\,\big((1-t_y)u_{i_1+1,i_2,i_3+1}+t_y u_{i_1+1,i_2+1,i_3+1}\big)\Big) \\
& +\lambda\Big(t_y(1-t_y)\,\big(u_{i_1,i_2-1,i_3}-2u_{i_1,i_2,i_3}+u_{i_1,i_2+1,i_3}\big) \\
& \qquad +t_z(1-t_z)\,\big(u_{i_1,i_2,i_3-1}-2u_{i_1,i_2,i_3}+u_{i_1,i_2,i_3+1}\big)\Big)\Bigg) = f_{i_1,i_2,i_3},
\end{aligned}
\tag{4.1}
$$

$i_1 = 2,3,\ldots N-1, i_2 = 1,2,\ldots N, i_3 = 1,2,\ldots N, N = 1/h.$

See Figure 2 for a representation of the discretization within the domain.

The inflow boundary conditions at $i_1 = 1$ are discretized

$$
\begin{aligned}
L^h u_{1,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (2+3\lambda)u_{1,i_2,i_3} \\
& -(2+2\lambda)\Big((1-\tfrac{t_z}{2})\,\big((1-\tfrac{t_y}{2})u_{0,i_2,i_3}+\tfrac{t_y}{2}u_{0,i_2-1,i_3}\big) \\
& \qquad +\tfrac{t_z}{2}\,\big((1-\tfrac{t_y}{2})u_{0,i_2,i_3-1}+\tfrac{t_y}{2}u_{0,i_2-1,i_3-1}\big)\Big) \\
& -\lambda\Big((1-t_z)\,\big((1-t_y)u_{2,i_2,i_3}+t_y u_{2,i_2+1,i_3}\big) \\
& \qquad +t_z\,\big((1-t_y)u_{2,i_2,i_3+1}+t_y u_{2,i_2+1,i_3+1}\big)\Big) \\
& +\lambda\Big(\tfrac{t_y}{2}(2-\tfrac{3t_y}{2})\,\big(u_{1,i_2-1,i_3}-2u_{1,i_2,i_3}+u_{1,i_2+1,i_3}\big) \\
& \qquad +\tfrac{t_z}{2}(2-\tfrac{3t_z}{2})\,\big(u_{1,i_2,i_3-1}-2u_{1,i_2,i_3}+u_{1,i_2,i_3+1}\big)\Big)\Bigg) = f_{1,i_2,i_3},
\end{aligned}
\tag{4.2}
$$

$i_2 = 1,2,\ldots N, i_3 = 1,2,\ldots N, N = 1/h; u_{0,i_2,i_3} = g(i_2 h, i_3 h).$

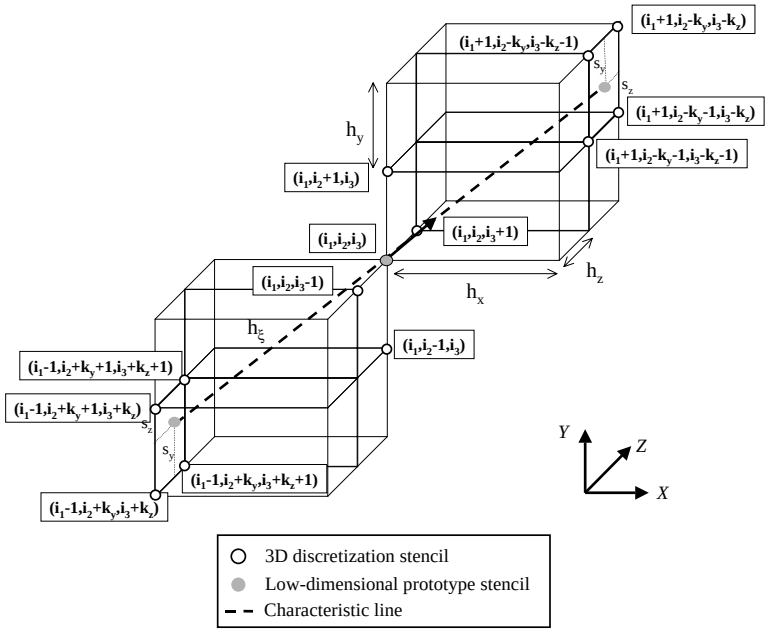

FIG. 3. *Rectangular-grid discretization stencil.*

The discrete operators near the outflow boundary are also derived from (3.1).

$$
\begin{aligned}
L^h u_{N,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (1+3\lambda)u_{N,i_2,i_3} \\
& -(1+\lambda)\Big((1-t_z)\,\big((1-t_y)u_{N-1,i_2,i_3}+t_y u_{N-1,i_2-1,i_3}\big) \\
& \qquad +t_z\,\big((1-t_y)u_{N-1,i_2,i_3-1}+t_y u_{N-1,i_2-1,i_3-1}\big)\Big) \\
& -2\lambda\Big((1-\tfrac{t_z}{2})\,\big((1-\tfrac{t_y}{2})u_{N+1,i_2,i_3}+\tfrac{t_y}{2}u_{N+1,i_2+1,i_3}\big) \\
& \qquad +\tfrac{t_z}{2}\,\big((1-\tfrac{t_y}{2})u_{N+1,i_2,i_3+1}+\tfrac{t_y}{2}u_{N+1,i_2+1,i_3+1}\big)\Big) \\
& +\lambda\Big(\tfrac{t_y}{2}(2-\tfrac{3t_y}{2})\,\big(u_{N,i_2-1,i_3}-2u_{N,i_2,i_3}+u_{N,i_2+1,i_3}\big) \\
& \qquad +\tfrac{t_z}{2}(2-\tfrac{3t_z}{2})\,\big(u_{N,i_2,i_3-1}-2u_{N,i_2,i_3}+u_{N,i_2,i_3+1}\big)\Big)\Bigg) = f_{N,i_2,i_3},
\end{aligned}
\tag{4.3}
$$

$i_2 = 1,2,\ldots N, i_3 = 1,2,\ldots N, N = 1/h.$

$$
\begin{aligned}
L^h u_{N+1,i_2,i_3} \equiv \tfrac{2}{h_\xi}\Bigg(& u_{N+1,i_2,i_3} - \Big((1-\tfrac{t_z}{2})\,\big((1-\tfrac{t_y}{2})u_{N,i_2,i_3}+\tfrac{t_y}{2}u_{N,i_2-1,i_3}\big) \\
& \qquad +\tfrac{t_z}{2}\,\big((1-\tfrac{t_y}{2})u_{N,i_2,i_3-1}+\tfrac{t_y}{2}u_{N,i_2-1,i_3-1}\big)\Big)\Bigg) = f_{N+1,i_2,i_3},
\end{aligned}
\tag{4.4}
$$

$i_2 = 1,2,\ldots N, i_3 = 1,2,\ldots N, N = 1/h.$

4.2. Rectangular-Grid Discretization. The coarse grids are 3-D Cartesian rectangular grids with aspect ratios $m_y = h_x/h_y$ and $m_z = h_x/h_z$, where h_x, h_y and h_z are the mesh sizes in the x, y and z

directions respectively. The basic discretization to the problem (2.2) on a coarse grid is defined as

$$
\begin{aligned}
L_b^{(h_x,h_y,h_z)} u_{i_1,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (1+2\lambda)u_{i_1,i_2,i_3} \\
& -(1+\lambda)\Big((1-s_z)\ \Big((1-s_y)u_{i_1-1,i_2+k_y,i_3+k_z} + s_y u_{i_1-1,i_2+(k_y+1),i_3+k_z}\Big) \\
& \qquad +s_z\ \Big((1-s_y)u_{i_1-1,i_2+k_y,i_3+(k_z+1)} + s_y u_{i_1-1,i_2+(k_y+1),i_3+(k_z+1)}\Big)\Big) \\
& \qquad -\lambda\Big((1-s_z)\ \Big((1-s_y)u_{i_1+1,i_2-k_y,i_3-k_z} + s_y u_{i_1+1,i_2-(k_y+1),i_3-k_z}\Big) \\
& \qquad +s_z\ \Big((1-s_y)u_{i_1+1,i_2-k_y,i_3-(k_z+1)} + s_y u_{i_1+1,i_2-(k_y+1),i_3-(k_z+1)}\Big)\Big) \\
& \qquad +\lambda\Big(s_y(1-s_y)\ \Big(u_{i_1,i_2-1,i_3} - 2u_{i_1,i_2,i_3} + u_{i_1,i_2+1,i_3}\Big) \\
& \qquad +s_z(1-s_z)\ \Big(u_{i_1,i_2,i_3-1} - 2u_{i_1,i_2,i_3} + u_{i_1,i_2,i_3+1}\Big)\Big)\Bigg) = f_{i_1,i_2,i_3}, \\
& i_1 = 2,3,\ldots N_x, i_2 = 1,2,\ldots N_y, i_3 = 1,2,\ldots N_z, N_x = 1/h_x, N_y = 1/h_y, N_z = 1/h_z,
\end{aligned}
\tag{4.5}
$$

where $h_\xi = h_x\sqrt{1+t_y^2+t_z^2}$, $k_y + s_y = -m_y t_y$, $k_z + s_z = -m_z t_z$, k_y, k_z are integers, $0 \le s_y, s_z < 1$. See Figure 3 for a pictorial explanation of the discretization stencil inside the domain. Note that on uniform grids ($m_y = m_z = 1$) the discretization (4.5) corresponds to (4.1) with $s_y = 1 - t_y$, $s_z = 1 - t_z$, and $k_y = k_z = -1$ provided $1 > t_y > 0$ and $1 > t_z > 0$.

The discretization in the inflow boundary cells ($i_1 = 1$) is defined similarly. Note, however, that the set of the non-alignment parameters is different.

$$
\begin{aligned}
L_b^{(h_x,h_y,h_z)} u_{1,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (2+3\lambda)u_{1,i_2,i_3} \\
& -2(1+\lambda)\Big((1-\tilde{s}_z)\ \Big((1-\tilde{s}_y)u_{0,i_2+\bar{k}_y,i_3+\bar{k}_z} + \tilde{s}_y u_{0,i_2+(\bar{k}_y+1),i_3+\bar{k}_z}\Big) \\
& \qquad +\tilde{s}_z\ \Big((1-\tilde{s}_y)u_{0,i_2+\bar{k}_y,i_3+(\bar{k}_z+1)} + \tilde{s}_y u_{0,i_2+(\bar{k}_y+1),i_3+(\bar{k}_z+1)}\Big)\Big) \\
& \qquad -\lambda\Big((1-s_z)\ \Big((1-s_y)u_{2,i_2-k_y,i_3-k_z} + s_y u_{2,i_2-(k_y+1),i_3-k_z}\Big) \\
& \qquad +s_z\ \Big((1-s_y)u_{2,i_2-k_y,i_3-(k_z+1)} + s_y u_{2,i_2-(k_y+1),i_3-(k_z+1)}\Big)\Big) \\
& \qquad +\tfrac{\lambda}{2}\Big(s_y(1-s_y)\ \Big(u_{1,i_2-1,i_3} - 2u_{1,i_2,i_3} + u_{1,i_2+1,i_3}\Big) \\
& \qquad +s_z(1-s_z)\ \Big(u_{1,i_2,i_3-1} - 2u_{1,i_2,i_3} + u_{1,i_2,i_3+1}\Big)\Big) \\
& \qquad +\lambda\Big(\tilde{s}_y(1-\tilde{s}_y)\ \Big(u_{1,i_2-1,i_3} - 2u_{1,i_2,i_3} + u_{1,i_2+1,i_3}\Big) \\
& \qquad +\tilde{s}_z(1-\tilde{s}_z)\ \Big(u_{1,i_2,i_3-1} - 2u_{1,i_2,i_3} + u_{1,i_2,i_3+1}\Big)\Big)\Bigg) = f_{1,i_2,i_3}, \\
& i_2 = 1,2,\ldots N_y, i_3 = 1,2,\ldots N_z, N_y = 1/h_y, N_z = 1/h_z,
\end{aligned}
\tag{4.6}
$$

where $\tilde{k}_y + \tilde{s}_y = -m_y\frac{t_y}{2}$ and $\tilde{k}_z + \tilde{s}_z = -m_z\frac{t_z}{2}$, $\tilde{k}_y$ and $\tilde{k}_z$ are integers, $0 \le \tilde{s}_y, \tilde{s}_z < 1$.

The outflow discretizations are defined as

$$
\begin{aligned}
L_b^{(h_x,h_y,h_z)} u_{N,i_2,i_3} \equiv \tfrac{1}{h_\xi}\Bigg(& (1+3\lambda)u_{N,i_2,i_3} \\
& -(1+\lambda)\Big((1-s_z)\ \big((1-s_y)u_{N-1,i_2+k_y,i_3+k_z} + s_y u_{N-1,i_2+(k_y+1),i_3+k_z}\big) \\
& \quad +s_z\ \big((1-s_y)u_{N-1,i_2+k_y,i_3+(k_z+1)} + s_y u_{N-1,i_2+(k_y+1),i_3+(k_z+1)}\big)\Big) \\
& -2\lambda\Big((1-\tilde{s}_z)\ \big((1-\tilde{s}_y)u_{N+1,i_2-\bar{k}_y,i_3-\bar{k}_z} + \tilde{s}_y u_{N+1,i_2-(\bar{k}_y+1),i_3-\bar{k}_z}\big) \\
& \quad +\tilde{s}_z\ \big((1-\tilde{s}_y)u_{N+1,i_2-\bar{k}_y,i_3-(\bar{k}_z+1)} + \tilde{s}_y u_{N+1,i_2-(\bar{k}_y+1),i_3-(\bar{k}_z+1)}\big)\Big) \\
& +\tfrac{\lambda}{2}\Big(s_y(1-s_y)\ \big(u_{N,i_2-1,i_3} - 2u_{N,i_2,i_3} + u_{N,i_2+1,i_3}\big) \\
& \quad +s_z(1-s_z)\ \big(u_{N,i_2,i_3-1} - 2u_{N,i_2,i_3} + u_{N,i_2,i_3+1}\big)\Big) \\
& +\lambda\Big(\tilde{s}_y(1-\tilde{s}_y)\ \big(u_{N,i_2-1,i_3} - 2u_{N,i_2,i_3} + u_{N,i_2+1,i_3}\big) \\
& \quad +\tilde{s}_z(1-\tilde{s}_z)\ \big(u_{N,i_2,i_3-1} - 2u_{N,i_2,i_3} + u_{N,i_2,i_3+1}\big)\Big)\Bigg) = f_{N,i_2,i_3},
\end{aligned}
\tag{4.7}
$$

$i_2 = 1, 2, \ldots N, i_3 = 1, 2, \ldots N, N = 1/h.$

$$
\begin{aligned}
L_b^{(h_x,h_y,h_z)} u_{N+1,i_2,i_3} \equiv \tfrac{2}{h_\xi}\Bigg(& u_{N+1,i_2,i_3} - \Big((1-\tilde{s}_z)\ \big((1-\tilde{s}_y)u_{N,i_2,i_3} + \tilde{s}_y u_{N,i_2-1,i_3}\big) \\
& \quad +\tilde{s}_z\ \big((1-\tilde{s}_y)u_{N,i_2,i_3-1} + \tilde{s}_y u_{N,i_2-1,i_3-1}\big)\Big)\Bigg) = f_{N+1,i_2,i_3},
\end{aligned}
\tag{4.8}
$$

$i_2 = 1, 2, \ldots N, i_3 = 1, 2, \ldots N, N = 1/h.$

4.3. Cross-Characteristic Interaction. The cross-characteristic interaction (CCI) introduced by a discrete operator (*inherent* CCI) can be quantitatively estimated by the coefficients of the lowest pure cross-characteristic derivatives appearing in the first differential approximation (FDA) (see [45]) to the discrete operator. In our model problem, the CCI appears only because of interpolation in the y-z plane. Therefore, the true CCI is actually determined by the FDA coefficients of ∂_{yy} and ∂_{zz}. The FDA to the uniform-grid discretization (4.1) taken for a characteristic component u ($\partial_y u \gg \partial_\xi u, \partial_z u \gg \partial_\xi u$) is given by

$$
FDA\Big(L^h\Big) = \partial_\xi - T_y^h h^2 \partial_{yy} - T_z^h h^2 \partial_{zz},
\tag{4.9}
$$

where

$$
\begin{aligned}
T_y^h &= \tfrac{t_y(1-t_y)}{2h\sqrt{1+t_y^2+t_z^2}}, \\
T_z^h &= \tfrac{t_z(1-t_z)}{2h\sqrt{1+t_y^2+t_z^2}},
\end{aligned}
\tag{4.10}
$$

for the inner points (discretizations (4.1) and (4.3)) and

$$
\begin{aligned}
T_y^h &= \tfrac{\frac{t_y}{2}(1-\frac{t_y}{2})}{h\sqrt{1+t_y^2+t_z^2}}, \\
T_z^h &= \tfrac{\frac{t_z}{2}(1-\frac{t_z}{2})}{h\sqrt{1+t_y^2+t_z^2}},
\end{aligned}
\tag{4.11}
$$

for the boundary points $i_1 = 1$ and $i_1 = N+1$ (discretizations (4.2) and (4.4)).

The first differential approximation to the coarse-grid discretization is

$$FDA\left(L^{(h_x,h_y,h_z)}\right) = \partial_\xi - T_y^{(h_x,h_y,h_z)} h_y^2 \partial_{yy} - T_z^{(h_x,h_y,h_z)} h_z^2 \partial_{zz} \tag{4.12}$$

where

$$\begin{aligned} T_y^{(h_x,h_y,h_z)} &= \frac{s_y(1-s_y)}{2h_x\sqrt{1+t_y^2+t_z^2}}, \\ T_z^{(h_x,h_y,h_z)} &= \frac{s_z(1-s_z)}{2h_x\sqrt{1+t_y^2+t_z^2}}, \end{aligned} \tag{4.13}$$

for the inner points (discretizations (4.5) and (4.7)) and

$$\begin{aligned} T_y^{(h_x,h_y,h_z)} &= \frac{\tilde{s}_y(1-\tilde{s}_y)}{h_x\sqrt{1+t_y^2+t_z^2}}, \\ T_z^{(h_x,h_y,h_z)} &= \frac{\tilde{s}_z(1-\tilde{s}_z)}{h_x\sqrt{1+t_y^2+t_z^2}}, \end{aligned} \tag{4.14}$$

for the boundary points (discretizations (4.6) and (4.8)).

Previous studies on different types of non-elliptic equations (see [3] and [4]) have shown that the main difficulty in constructing an efficient multigrid solver is a poor coarse-grid approximation to the fine-grid characteristic error components. It was observed that a coarse-grid operator defined on a grid built by full coarsening unavoidably introduces a too strong CCI. On the other hand, a narrow discretization (4.5) on a semicoarsened grid (only the x-directional mesh size is doubled) results in a coarse-grid CCI that is lower than required. However, operator (4.5) on the semicoarsened grid can be supplied with additional terms (*explicit* CCI), so that the *total* CCI would be exactly the same as on the fine grid. Thus, one could derive an appropriate coarse-grid operator by comparing the FDA (4.12) with the target-grid FDA (4.9)

$$\begin{aligned} L^{(h_x,h_y,h_z)} u_{i_1,i_2,i_3} \equiv L_b^{(h_x,h_y,h_z)} \quad & -A_y(u_{i_1,i_2-1,i_3} - 2u_{i_1,i_2,i_3} + u_{i_1,i_2+1,i_3}) \\ & -A_z(u_{i_1,i_2,i_3-1} - 2u_{i_1,i_2,i_3} + u_{i_1,i_2,i_3+1}), \end{aligned} \tag{4.15}$$

where

$$\begin{aligned} A_y &= T_y^h - T_y^{(h_x,h_y,h_z)}, \\ A_z &= T_z^h - T_z^{(h_x,h_y,h_z)}. \end{aligned} \tag{4.16}$$

The idea is that the characteristic error components oscillating highly in the cross-characteristic directions are eliminated in relaxation, while the smooth characteristic components are well approximated on the coarse grids with operator (4.15).

4.4. Uniform Cross-Characteristic Interaction. In general problems, where the non-alignment parameters are changed from node to node (e.g., because of variable coefficients or grid stretching), the target-grid CCI may vary as well. It has been shown in [3] that for vertex-centered formulations, treatment of a variable CCI is not a problem. The fact that the grids are nested ensures the quality of the coarse-grid correction to the characteristic error components. The situation is different for cell-centered grids. In the case of (near) diagonal alignment, the inherent CCI vanishes on all the grids, and therefore all the characteristic error components must be reduced in the coarse-grid correction. For non-nested grids, however, the characteristic lines passing through the grid nodes on different grids are parallel but do not coincide (see Figure 4). This implies that the fine-grid characteristic components oscillating highly in the cross-characteristic direction cannot be approximated on the coarse grids. The proposed cure is to supply the target grid discretization with explicit CCI terms (similar to the coarse-grid explicit CCI terms) ensuring

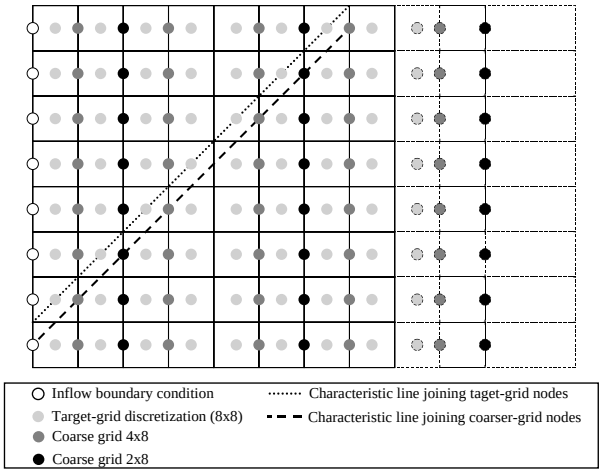

FIG. 4. *Characteristic lines in different grid levels.*

that the total CCI is never vanishes. In the multigrid solver reported in this paper, we impose a uniform CCI, i.e., the total CCI in each discretization node satisfies

$$T_{\text{total}} = \frac{0.19}{h}, \tag{4.17}$$

where $h = h_y = h_z$ is the same on all the grids. This value of T_{total} approximately corresponds to the maximum uniform-grid inherent CCI of the discretization (4.2). The maximum is taken over different values of the non-alignment parameters t_y and t_z.

5. The Multigrid Method. The proposed multigrid method for solving the convection equation employs semicoarsening and narrow coarse-grid discretization schemes supplied with explicit terms (which are discrete approximations to $h_y\partial_{yy}$ and $h_z\partial_{zz}$ with suitable coefficients) to maintain on all the grids the same uniform CCI. This construction ensures that all the *characteristic error components* are eliminated fully by the coarse-grid correction. The *non-characteristic error components* must be reduced in relaxation. Successive semicoarsening implies a fast decrease in the *inherent* CCI on coarse grids and, hence, a fast increase in the weight of the explicit terms in the coarse-grid operators (since the total CCI remains fixed). Eventually, on coarse grids, the cross-characteristic coupling defined by the explicit terms becomes dominant. Thus, plane smoothers should be applied to reduce the non-characteristic error components.

The tested semicoarsening multilevel $V(\nu_1, \nu_2)$ cycle consists of a four-color plane-implicit relaxation scheme, an upwind restriction operator for the residual transfer, and a linear prolongation operator for the coarse-grid correction. On each level, except the coarsest one where the problem is directly solved, ν_1 relaxation sweeps are performed before transferring residuals to the coarse grid and ν_2 sweeps are performed after receiving coarse-grid corrections.

5.1. Plane Relaxation Scheme. The plane relaxation four-color scheme employed in the multigrid cycle is derived from the one-dimensional scheme described in Section 3.1. Now, the number of colors determines the order of relaxing the y-z grid planes. The planes with the x-axis coordinates $i_1 = 1 + jp$, $(j \in Z)$ are relaxed first, then the planes with $i_1 = 2 + jp$, and so on; the last planes to be relaxed are those with $i_1 = 4 + jp$. In the plane relaxation scheme, all the solution values on the same y-z grid plane are

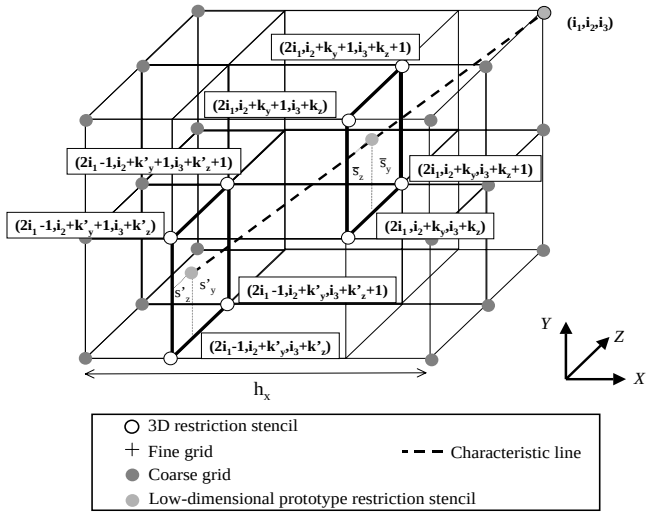

FIG. 5. *Full-dimensional stencil of the restriction operator.*

updated altogether. Simultaneous solution of all the equations centered on a plane would reduce residuals in this plane to zero. However, an exact solution is not needed [23, 37]; the planes are solved approximately by a single 2-D V(1,1) multigrid cycle with an alternating-line smoother. This inexact solution of the planes does not decrease the convergence of the multigrid algorithm.

5.2. Intergrid Transfers. *The residual transfer* to the semicoarsened grid is given by

$$
\begin{aligned}
R_{i_1,i_2,i_3} = \tfrac{1}{2}\Bigg(\Big(& (1-\bar{s}_z) \quad ((1-\bar{s}_y) r_{2i_1,\, i_2+\bar{k}_y,\, i_3+\bar{k}_z} + \bar{s}_y r_{2i_1,\, i_2+\bar{k}_y+1,\, i_3+\bar{k}_z}) \\
& +\bar{s}_z \quad ((1-\bar{s}_y) r_{2i_1,\, i_2+\bar{k}_y,\, i_3+\bar{k}_z+1} + \bar{s}_y r_{2i_1,\, i_2+\bar{k}_y+1,\, i_3+\bar{k}_z+1}) \Big) \\
+ \Big(& (1-s'_z) \quad ((1-s'_y) r_{2i_1-1,\, i_2+k'_y,\, i_3+k'_z} + s'_y r_{2i_1-1,\, i_2+k'_y+1,\, i_3+k'_z}) \\
& +s'_z \quad ((1-s'_y) r_{2i_1-1,\, i_2+k'_y,\, i_3+k'_z+1} + s'_y r_{2i_1-1,\, i_2+k'_y+1,\, i_3+k'_z+1}) \Big) \Bigg)
\end{aligned}
\tag{5.1}
$$

where $r_{i_1,i_2,i_3} = f_{i_1,i_2,i_3} - L^{(h_x,h_y,h_z)} u_{i_1,i_2,u_3}$ is the fine-grid residual function, and R_{i_1,i_2,i_3} is the coarse-grid residual function. The non-alignment parameters for the restriction operator are defined as $\bar{k}_y + \bar{s}_y = -\frac{m_y}{4} t_y$, $\bar{k}_z + \bar{s}_z = -\frac{m_y}{4} t_z$, $k'_y + s'_y = -\frac{3m_y}{4} t_y$, and $k'_z + s'_z = -\frac{3m_z}{4} t_z$, where $\bar{k}_y$, $\bar{k}_z$,k'_y, and k'_z are integers, $0 \le \bar{s}_y, \bar{s}_z, s'_y, s'_z < 1$, and m_y and m_z are the aspect ratios in the coarse grid. See Figure 5 for a representation of the restriction operator.

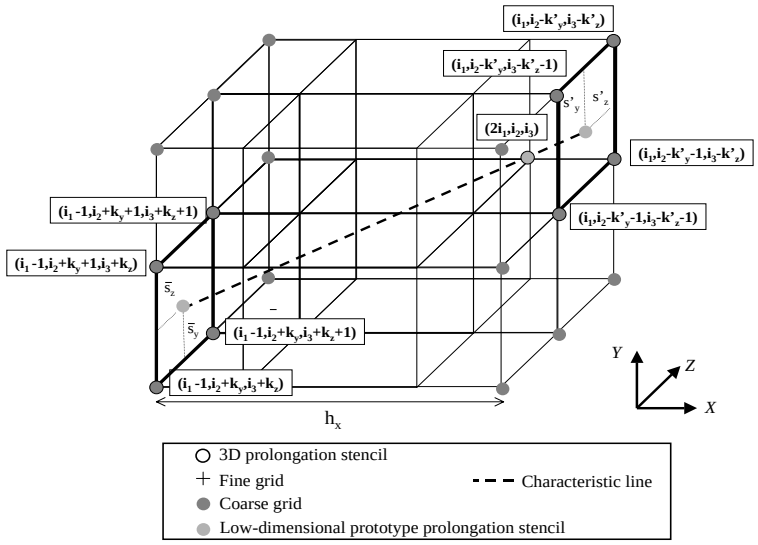

FIG. 6. *Full-dimensional stencil of the prolongation operator.*

The coarse-grid correction operator is a linear interpolation defined by the following formulae. For the nodes with even x-directional indices,

$$
\begin{aligned}
v_{2i_1,i_2,i_3} = \tfrac{1}{4}\Bigg(\Bigg(&(1-s'_z) \quad \Big((1-s'_y)V_{i_1-1,i_2+k'_y,i_3+k'_z} + s'_y V_{i_1-1,i_2+k'_y+1,i_3+k'_z}\Big)\\
&+s'_z \quad \Big((1-s'_y)V_{i_1-1,i_2+k'_y,i_3+k'_z+1} + s'_y V_{i_1-1,i_2+k'_y+1,i_3+k'_z+1}\Big)\Bigg)\\
+3\Bigg(&(1-\bar{s}_z) \quad \Big((1-\bar{s}_y)V_{i_1,i_2-\bar{k}_y,i_3-\bar{k}_z} + \bar{s}_y V_{i_1,i_2-\bar{k}_y-1,i_3-\bar{k}_z}\Big)\\
&+\bar{s}_z \quad \Big((1-\bar{s}_y)V_{i_1,i_2-\bar{k}_y,i_3-\bar{k}_z-1} + \bar{s}_y V_{i_1,i_2-\bar{k}_y-1,i_3-\bar{k}_z-1}\Big)\Bigg)\Bigg),
\end{aligned}
\tag{5.2}
$$

and for the nodes with odd x-directional indices,

$$
\begin{aligned}
v_{2i_1-1,i_2,i_3} = \tfrac{1}{4}\Bigg(3\Bigg(&(1-\bar{s}_z) \quad \Big((1-\bar{s}_y)V_{i_1-1,i_2+\bar{k}_y,i_3+\bar{k}_z} + \bar{s}_y V_{i_1-1,i_2+\bar{k}_y+1,i_3+\bar{k}_z}\Big)\\
&+\bar{s}_z \quad \Big((1-\bar{s}_y)V_{i_1-1,i_2+\bar{k}_y,i_3+\bar{k}_z+1} + \bar{s}_y V_{i_1-1,i_2+\bar{k}_y+1,i_3+\bar{k}_z+1}\Big)\Bigg)\\
+\Bigg(&(1-s'_z) \quad \Big((1-s'_y)V_{i_1,i_2-k'_y,i_3-k'_z} + s'_y V_{i_1,i_2-k'_y-1,i_3-k'_z}\Big)\\
&+s'_z \quad \Big((1-s'_y)V_{i_1,i_2-k'_y,i_3-k'_z-1} + s'_y V_{i_1,i_2-k'_y-1,i_3-k'_z-1}\Big)\Bigg)\Bigg),
\end{aligned}
\tag{5.3}
$$

where V is the coarse-grid solution and v is the correction to the fine-grid solution approximation. The non-alignment parameters for the prolongation operator are similar to those defined for the restriction operator. See Figure 6 for a pictorial representation of the prolongation operator.

5.3. Multigrid Cycle. In the full dimension, a multigrid $V(\nu_1,\nu_2)$ cycle employing semicoarsening can be defined as the following five steps

Step 1: Prerelaxation sweeps. The current approximation is improved by ν_1 plane-implicit relaxation sweeps.

Step 2: Residual transfer. The coarse-grid approximation to the fine-grid residual function is calculated by means of (5.1).
Step 3: Coarse-grid solution. The coarse-grid problem is solved by the recursive call of the same $V(\nu_1, \nu_2)$ cycle. On the coarsest grid, the problem is solved directly.
Step 4: Coarse-grid correction. The coarse-grid solution is interpolated by (5.2), (5.3) to the fine grid. The current fine-grid approximation is corrected.
Step 5: Postrelaxation sweeps. The current fine-grid approximation is improved by ν_2 plain-implicit relaxation sweeps.

5.4. Implementation Aspects. The code has been implemented to deal with structured rectangular grids with stretching. A generalization of discretization (4.15) and intergrid transfers (5.1), (5.2), (5.3) is applied. The non-alignment parameters are different at each grid node. We could recompute and store the non-alignment parameters in the memory so we do not have to recompute them each time the residual, the metrics or the intergrid transfers are computed. However, such an approach increases the memory requirements of the code.

Before applying the multigrid cycles, the metrics and the non-alignment parameters for the discretization in the left, central and right planes are calculated and stored for each node and grid. However, the non-alignment parameters for the intergrid transfers are recomputed each time the operators are applied. This solution provides a trade-off between computing and wasted memory.

TABLE 6.1
Asymptotic/geometric-average convergence rate for the solver-case combinations.

	64^3				128^3			
	case 1	case 2	case 3	case 4	case 1	case 2	case 3	case 4
solver 1	0.07/0.06	0.06/0.05	0.09/0.07	0.04/0.04	0.08/0.07	0.06/0.05	0.10/0.08	0.04/0.04
solver 2	0.16/0.18	0.11/0.10	0.25/0.28	0.08/0.07	0.24/0.29	0.17/0.19	0.39/0.44	0.10/0.08

6. Numerical Results. The inflow boundary conditions for the test problems were chosen so that the function $U(x, y, z) = \cos(\omega(y + z - (t_y + t_z)x))$ is the exact continuous solution of the homogeneous ($f_{i_1,i_2,i_3} = 0$) problem (2.2). The initial approximation was interpolated from the solution on the previous coarse grid.

The frequencies $\omega = 8\pi$ for a 64^3 grid and $\omega = 16\pi$ for a 128^3 grid were chosen to reduce the total computational time exploiting periodicity and to provide a reasonable accuracy in approximating the true solution of the differential equation. Two cycles, with (*solver 1*) and without (*solver 2*) explicit CCI terms in coarse-grid operators, were compared on 64^3 and 128^3 uniform grids for the non-alignment parameters

- $t_y = 0.2$ $t_z = 0.2$ (*case 1*)
- $t_y = 0.98$ $t_z = 0.2$ (*case 2*)
- $t_y = 0.5$ $t_z = 0.0$ (*case 3*)
- $t_y = 0.98$ $t_z = 0.98$ (*case 4*)

Table 6.1 contains the asymptotic and geometric-average convergence rates and Figure 7 shows the residual history versus work units; the work unit is the computer-operation count in the target-grid residual evaluation.

The plane solver used in the 3-D smoothing procedure is a robust two-dimensional (2-D) multigrid V(1,1)-cycle employing full coarsening and alternating-line smoothers; the approximate 2-D solution obtained after one 2-D cycle is sufficient to provide robustness and good convergence rates in the 3-D solvers. The

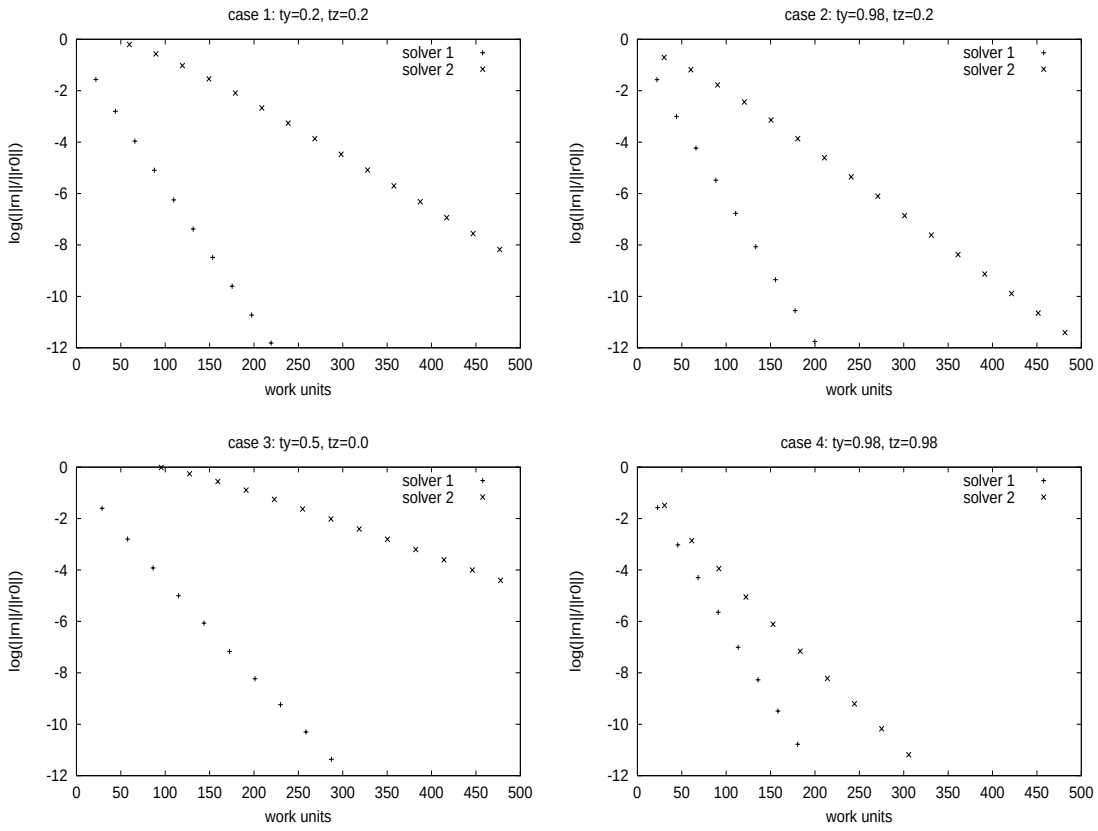

FIG. 7. *Residual versus work units for $V(1,1)$ cycles with semicoarsening: Solver 1 (with explicit CCI terms) and Solver 2 (without explicit CCI terms) for a 128^3 grid and $w = 16\pi$.*

combination of semicoarsening, the four-color plane-implicit smoother, and the introduction of explicit CCI terms in grid discretizations (*solver 1*) yields a multigrid solver with fast grid-independent convergence rates for any angles of non-alignment. The algorithm without explicit CCI terms (*solver 2*) presents a worse and grid-dependent convergence rates.

Stretched grids are commonly used in CFD grid generation to pack points into regions with large solution gradients while avoiding an excess of points in more benign regions, and to capture viscous effects in the boundary layers. The stretching of the grid in a given direction is determined by the stretching geometric factor β (quotient between two consecutive space steps, $h_k = \beta h_{k-1}$), which produces aspect ratios up to β^{N-1} if the three directions are equally stretched. In order to study the effect of stretching on the behavior of the solvers, the grids were stretched using a geometric factor $\beta = 1.1$ (Figure 8), which produces aspect ratios up to order 10^5 on 128^3 grids.

Two multigrid cycles, with CCI correction in every grid operator (*solver 1*) and without explicit CCI terms (*solver 2*) were compared on 64^3 and 128^3 stretched grids for the non-alignment parameters

- $t_y = 0.2$ $t_z = 0.2$ (*case 1*)
- $t_y = 0.98$ $t_z = 0.2$ (*case 2*)
- $t_y = 0.5$ $t_z = 0.0$ (*case 3*)
- $t_y = 0.98$ $t_z = 0.98$ (*case 4*)

Table 6.2 contains the asymptotic and geometric-average convergence rates and Figure 9 shows the

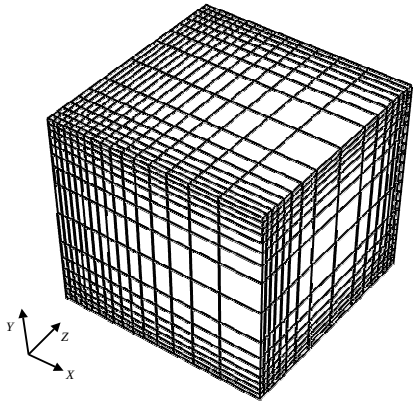

FIG. 8. 16^3 *stretched grid with geometric factor* $\beta = 1.1$.

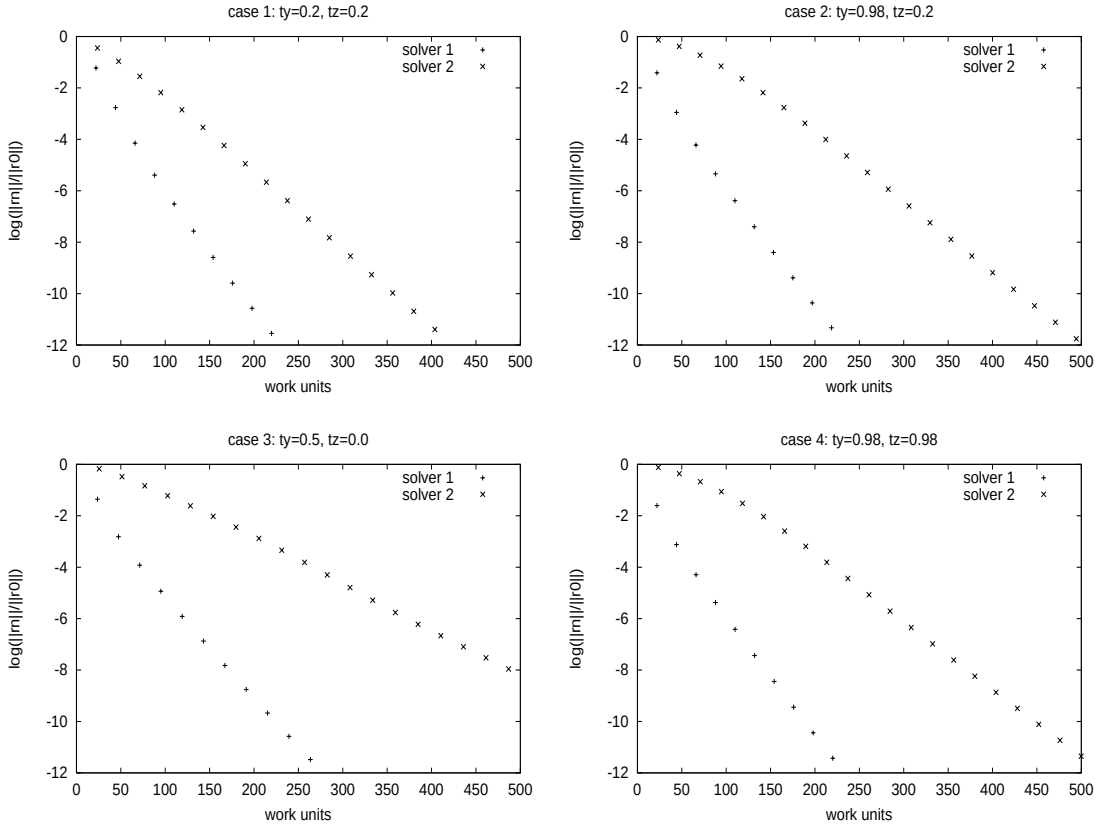

FIG. 9. *Residual versus work units for* $V(1,1)$ *cycles with semicoarsening: Solver 1 (with explicit CCI terms) and Solver 2 (without explicit CCI terms) for a stretched* 128^3 *grid and* $w = 16\pi$.

residual history versus work units. The multigrid cycle without explicit CCI terms (*solver 2*) always presents a grid-dependent convergence rate. The combination of semicoarsening, the four-color plane-implicit smoother, and a uniform CCI correction in all grids (*solver 1*) yields a multigrid solver with fast grid-independent convergence rates for any angles of non-alignment and grid stretching.

TABLE 6.2

Asymptotic/geometric-average convergence rate for the solver-case combinations with grid stretching.

	64^3				128^3			
	case 1	case 2	case 3	case 4	case 1	case 2	case 3	case 4
solver 1	0.10/0.07	0.10/0.07	0.11/0.08	0.10/0.07	0.10/0.07	0.10/0.07	0.12/0.09	0.10/0.07
solver 2	0.13/0.13	0.23/0.18	0.23/0.20	0.22/0.18	0.20/0.21	0.23/0.27	0.30/0.36	0.24/0.28

7. Efficient Solution of the Convection-Diffusion Operator. In this Section we will study the 3-D constant-coefficient convection-diffusion equation

$$\epsilon \Delta U = F(x,y,z) + \Big(\bar{a}\cdot\nabla\Big)U, \tag{7.1}$$

where $\bar{a} = (a_1, a_2, a_3)$ is a given vector and ϵ is a positive scalar. The solution $U(x,y,z)$ is a differentiable function defined on the unit square $(x,y,z) \in [0,1]\times[0,1]\times[0,1]$.

Equation (7.1) can be rewritten as

$$\epsilon(\partial_{xx}U + \partial_{yy}U + \partial_{zz}U) = F(x,y,z) + |\bar{a}|\partial_\xi U, \tag{7.2}$$

and with the additional streamwise dissipation as

$$\epsilon(\partial_{xx}U + \partial_{yy}U + \partial_{zz}U) = F(x,y,z) + |\bar{a}|(\partial_\xi U - \lambda h_\xi \partial_{\xi\xi}U), \tag{7.3}$$

where $|\bar{a}| = \sqrt{a_1^2 + a_2^2 + a_3^2}$, and $\xi = \frac{x + t_y y + t_z z}{\sqrt{1+t_y^2+t_z^2}}$ is a variable along the characteristic of the convective part. Equation (7.1) is subject to Dirichlet boundary conditions at the inflow boundary $x = 0$, Neumann boundary conditions at the outflow boundary $x = 1$ and periodic conditions in the y and z directions

$$U(0,y,z) = g(y,z), U_x(1,y,z) = 0, U(x,y,z) = U(x,y+1,z), U(x,y,z) = U(x,y,z+1), \tag{7.4}$$

where $g(y,z)$ is a given function.

For the diffusive part, the left-hand side of Eq. (7.3), on stretched grids, each cell can have a different aspect ratio. So its discretization is given by the following general discrete operator:

$$\begin{aligned}
&\frac{2\epsilon}{hx_{i_1}}\Big(\frac{u_{i_1-1,i_2,i_3}}{hx_{i_1}+hx_{i_1-1}} - \Big(\frac{u_{i_1,i_2,i_3}}{hx_{i_1}+hx_{i_1+1}} + \frac{u_{i_1,i_2,i_3}}{hx_{i_1}+hx_{i_1-1}}\Big) + \frac{u_{i_1+1,i_2,i_3}}{hx_{i_1}+hx_{i_1+1}}\Big) + \\
&\frac{2\epsilon}{hy_{i_2}}\Big(\frac{u_{i_1,i_2-1,i_3}}{hy_{i_2}+hy_{i_2-1}} - \Big(\frac{u_{i_1,i_2,i_3}}{hy_{i_2}+hy_{i_2+1}} + \frac{1}{hy_{i_2}+hy_{i_2-1}}\Big) + \frac{u_{i_1,i_2+1,i_3}}{hy_{i_2}+hy_{i_2+1}}\Big) + \\
&\frac{2\epsilon}{hz_{i_3}}\Big(\frac{u_{i_1,i_2,i_3-1}}{hz_{i_3}+hz_{i_3-1}} - \Big(\frac{u_{i_1,i_2,i_3}}{hz_{i_3}+hz_{i_3+1}} + \frac{u_{i_1,i_2,i_3}}{hz_{i_3}+hz_{i_3-1}}\Big) + \frac{u_{i_1,i_2,i_3+1}}{hz_{i_3}+hz_{i_3+1}}\Big); \\
&\qquad i_1 = 1,...,N_x, i_2 = 1,...,N_y, i_3 = 1,...,N_z.
\end{aligned} \tag{7.5}$$

We use the same multigrid method as the one previously applied for the convection operator (four-color plane-implicit relaxation scheme combined with semicoarsening). For the diffusion operator, however, symmetric (rather than upwind biased) versions of the intergrid transfers are preferable. For example, the restriction operator

$$R_{i_1,i_2,i_3} = \frac{1}{2}\Big(r_{2i_1,i_2,i_3} + r_{2i_1+1,i_2,i_3}\Big), \tag{7.6}$$

and the prolongation operator

$$\begin{cases} v_{2i_1,i_2,i_3} &= \frac{1}{4}\Big(V_{i_1-1,i_2,i_3} + 3V_{i_1,i_2,i_3}\Big), \\ v_{2i_1-1,i_2,i_3} &= \frac{1}{4}\Big(3V_{i_1-1,i_2,i_3} + V_{i_1,i_2,i_3}\Big), \end{cases} \tag{7.7}$$

(compare with (5.1)-(5.3)) were successfully used in [37] to build a robust multigrid method based on semi-coarsening combined with plane-implicit smoothing for solving the anisotropic diffusion operator (7.5). Volume weighted versions of (7.6) and (7.7) are required for stretched grids.

The intergrid transfers for the convection-diffusion equation are implemented as a weighted average of the operators (5.1) and (7.6) for the restriction operator and the operators (5.2)-(5.3) and (7.7) for the prolongation operator. Specifically, the operators (7.6) and (7.7) are taken with the weighting $\frac{\epsilon}{|\bar{a}|h}$, while the weighting of the operators (5.1)-(5.3) is $\left(1 - \frac{\epsilon}{|\bar{a}|h}\right)$. If $\frac{\epsilon}{|\bar{a}|h} \geq 1$, only symmetric operators are used.

Again, the inflow boundary conditions for the test problems were chosen so that $g(y,z) = \cos(\omega(y+z))$ and $F_{i_1,i_2,i_3} = 0$. The initial approximation was interpolated from the solution on the previous coarse grid and the frequencies $\omega = 8\pi$ for a 64^3 grid and $\omega = 16\pi$ for a 128^3 grid were chosen. Two multigrid cycles, with (*solver 1*) and without (*solver 2*) explicit CCI terms in operators, were compared on 64^3 and 128^3 uniform grids for the non-alignment parameters $t_y = t_z = 0.5$ and the following diffusive values:

- *case 1:* $\epsilon \approx 10^{-4}$, *very small diffusion*, the problem is convective dominated
- *case 2:* $\epsilon \approx |\bar{a}|h$, *small diffusion*, the physical diffusion is of the same order as the inherent numerical dissipation in the target grid
- *case 3:* $\epsilon \approx 6|\bar{a}|h$, *intermediate diffusion*, the physical diffusion is larger than the inherent numerical dissipation in the target grid, but smaller than a possible inherent numerical dissipation in some coarse grids
- *case 4:* $\epsilon \approx 1.0$, *large diffusion*, the physical diffusion is O(1)
- *case 5:* $\epsilon \approx 10^3$, *very large diffusion*, the problem is diffusive dominated

Tables 7.1 and 7.2 contain the asymptotic and geometric-average convergence rates for V(1,1) cycles on uniform and stretched grids respectively. Both the solvers present fast grid-independent convergence rates for problems with a non-negligible diffusive part (*cases 2-5*). However, as was previously shown in Section 6, the convergence rate becomes grid dependent without explicit CCI terms for convection dominated problems (*case 1*).

TABLE 7.1

Asymptotic/geometric-average convergence rate for the solver-case combinations for the convection-diffusion problem.

	64^3				
	case 1	case 2	case 3	case 4	case 5
solver 1	0.06/0.06	0.10/0.09	0.09/0.07	0.07/0.05	0.05/0.05
solver 2	0.27/0.31	0.09/0.08	0.09/0.07	0.07/0.05	0.05/0.05
	128^3				
	case 1	case 2	case 3	case 4	case 5
solver 1	0.10/0.08	0.10/0.09	0.09/0.07	0.07/0.05	0.05/0.05
solver 2	0.42/0.46	0.09/0.07	0.09/0.07	0.07/0.05	0.05/0.05

8. Parallel Implementation. The main advantage of the multigrid algorithm proposed in this report is its parallel potential. We now describe a parallel implementation of the algorithm, based on MPI, and its efficiency on two different parallel systems; a Cray T3E-900 (T3E) and an SGI Origin 2000 (O2K).

The test problem chosen in the experiments carried out in this section is the convection-diffusion equation (7.3) with boundary conditions (7.4), so that $g(y,z) = \cos(\omega(y+z))$, and $F_{i_1,i_2,i_3} = 0$. We have selected frequencies $\omega = 8\pi$ and 16π for 64^3 and 128^3 grids respectively, with $t_y = t_z = 0.5$ as non-alignment parameters and $\epsilon \approx 10^{-4}$ as the diffusive value (*case 1* in the previous Section). The initial approximation

TABLE 7.2

Asymptotic/geometric-average convergence rate for the solver-case combinations for the convection-diffusion problem with grid stretching.

	64^3				
	case 1	case 2	case 3	case 4	case 5
solver 1	0.11/0.08	0.15/0.12	0.15/0.10	0.13/0.09	0.16/0.09
solver 2	0.24/0.21	0.15/0.13	0.15/0.10	0.13/0.09	0.16/0.09
	128^3				
	case 1	case 2	case 3	case 4	case 5
solver 1	0.12/0.09	0.15/0.11	0.15/0.11	0.14/0.09	0.14/0.09
solver 2	0.30/0.38	0.15/0.13	0.15/0.13	0.14/0.09	0.14/0.09

was interpolated from the solution on the previous coarse grid.

8.1. Cray T3E and SGI Origin 2000 Architectures. The T3E that we have used in this study is based on the DEC Alpha 21164 (DEC Alpha EV5) processor running at 450 MHz. This processor has two levels of caching on-chip (8-KB first-level instructions and data caches, and a unified 96-KB second-level cache). However, unlike other systems, the EV5 contains no board-level cache. The nodes (or processors in this case) are connected by means of a 3-D torus network, whose links provide a raw bandwidth of 600 MB/s in each direction [5, 1]. Nevertheless, the effective communication bandwidth obtained with the classical ping-pong test is approximately 300 MB/s (around half of the peak) [32].

The O2K employed consists of 32 MIPS R10000 processors running at 250 MHz. This processor has 32-KB primary data and instruction caches on chip, but its unified 4-MB L2 cache is external. Unlike the T3E, each O2K node consists of two processors connected by a system bus (SysAD bus). Along with the processors, each node has a portion of the shared main memory (512 MB in our system), a directory for cache coherence, an I/O interface, and the Hub, which is the combined communication/coherence controller and network interface. The network is based on a flexible switch called SPIDER. Two nodes (four processors) are connected to each switch through their Hubs. Systems with 32 processors, such as the one that we have used, are created from a three-dimensional hypercube of switches where only five of the six links on each router are used. The peak bandwidth of the bus that connects the two processors and the Hub's connections to memory and routers is 780 MB/s. However, at user level, the actual effective bandwidth between processors is much lower than the peak, due to the cache-coherency protocol and other overheads [5, 20]. Using MPI, the maximum achievable bandwidth is only around 140 MB/s [32]. We should mention that in this system, we have combined MPI with the SGI dplace tool in order to disable the automatic page migration, which is not useful in MPI programs, and also to specify the distributions of threads onto memories since both these characteristics improve performance [39].

8.2. Parallel Strategies. Basically, one can adopt two different strategies to get a parallel implementation of a multigrid method: domain decompositions combined with multigrid (DD-MG) and grid partitioning (MG-DD).

The DD-MG approach consists in applying a domain decomposition on the finest grid, and using a multigrid method inside each block. These kind of methods are often considered with finite element discretization since they are easier to implement and can be directly applied to general multi-block grids. From an architectural point of view, they also imply fewer communications since they are only required on the finest grid. However, they lead to algorithms which are numerically different to the sequential version and

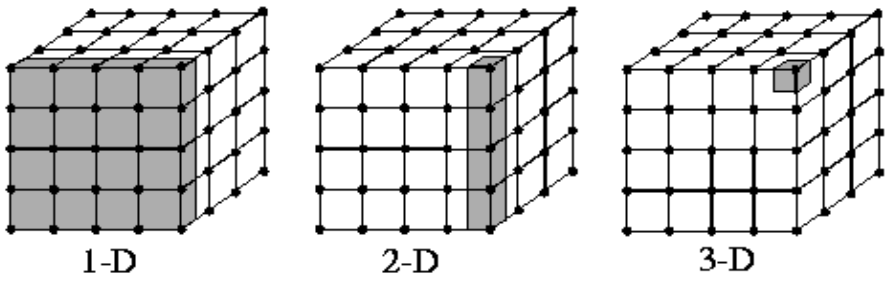

FIG. 10. *1-D, 2-D and 3-D decompositions of a three-dimensional domain*

have a negative impact on the convergence rate [40].

We have limited this research to the MG-DD technique. In this approach, multigrid is used to solve the problem in the whole grid, i.e. domain decomposition is applied on each level. Therefore, it implies more communication overheads since data exchange is required on every level. However, unlike DD-MG approaches, it retains the convergence rate of the sequential algorithm [24]. Regarding multi-block grids, we should mention that an adequate hybrid approach, where the V-cycle is applied throughout the entire domain while the smoothers are performed inside each block (domain decomposition for the smoother), has been proposed in [21, 22].

As is well known, for three-dimensional domains three different decompositions are possible (see Figure 10). Common grid transfer operators such as linear interpolation and full weighted restrictors are parallel by nature. Hence, regarding these operators, it does not matter which decomposition is chosen. However, 2-D and 3-D decompositions require a parallel plane solver for the smoothing process. Therefore, as we have employed an alternating-line smoother in the plane solver, such partitionings need a parallel tridiagonal solver. This problem has been widely studied (see for example [19, 16, 26, 18, 11]). More specifically, we have studied, in [12, 13], tridiagonal solvers in the context of an alternating line process such as the plane smoother of our multigrid algorithm. In particular, we have revised a Pipelined Gaussian Elimination method [31, 30], Wang's algorithm [43], and several methods based on data transpositions [13]. The experimental results obtained on a T3E-900 with up to 512 processors have been quite satisfactory for large and even moderate 2-D problem sizes (from a 1024^2 problem size), with parallel efficiencies in the range 0.7 to 0.9. However, current memory limitations prevent us from solving 3-D problems whose corresponding 2-D planes are big enough to obtain reasonable efficiencies on medium-sized parallel computers.

We have taken the decision, based on these results, to use a 1-D data decomposition in the semi-coarsened direction, since it does not require a parallel 2-D solver. Considering that our code was written in C language, where 3-D matrices are stored in a row-ordered (x,y,z)-array, YZ and XZ-planes represent continuous data (except for the gap between different z-columns), while XY-planes reference data that have a stride of twice the number of elements in dimension Z (see Figure 11). Consequently, x-semicoarsening and y-semicoarsening are more efficient than z-semicoarsening, because they exhibit a better spatial locality. For the same reasons, x and y-semicoarsening are also the best choice in terms of message passing, since the effective communication bandwidth on the systems under study also presents a significant dependence on the spatial locality of the messages [32, 33]. Just to give a few results, using the Apprentice profiling tool on the T3E, we have recorded that for a 64^3 problem size on a two-processor simulation the time spent

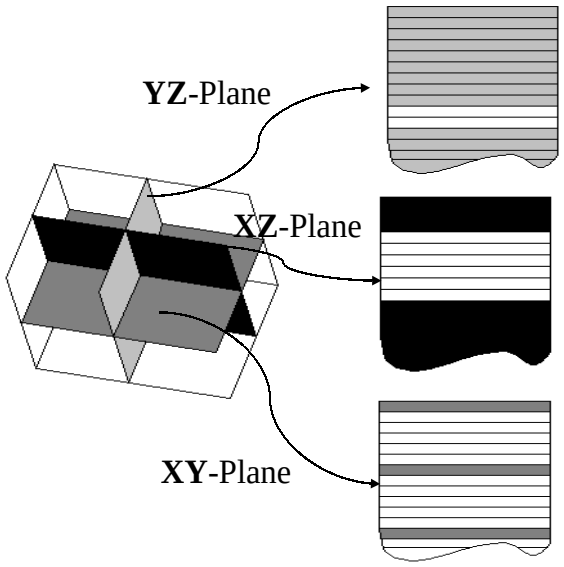

FIG. 11. *Access patterns for different planes*

performing data cache operations using z-partitioning is about 17% larger than with the x-partitioning [32]. All the experimental results presented in this paper have been obtained using an x-direction partitioning, that is, x-semicoarsening combined with YZ-plane smoothers.

Regarding data locality, we should mention that in order to improve the data locality of iterative methods some authors have successfully employed different techniques, such as data access transformations (loop interchange, loop fusion, loop blocking, and prefetching) and data layout transformations (array padding and array merging)[36, 35, 10]. Our codes have been compiled using aggressive compiler options (Ofast=ip27 on the O2K and O3 on the T3E), but we have not employed any data transformations to optimize cache reuse. However, plane smoothers allow a better exploitation of the temporal locality compared to common point smoothers, which have to perform global sweeps through data sets that are too large to fit in the cache.

8.3. Critical Level Problem. Although 1-D decompositions have no need for a parallel plane smoother, they also present some drawbacks caused by the need to solve exactly the linear system of equations on the coarsest grid. In sequential algorithms, the coarsest level is usually chosen as coarse as possible to reduce the computational cost. However, in the parallel versions, this decision may cause some processors to remain idle on the coarsest grids. To clarify this problem, it is convenient to define the multigrid critical level as the level L where the following condition is satisfied:

$$\frac{N(L)}{coef \cdot P} = 1 \tag{8.1}$$

where $N(L)$ is the local number of cells per side of level L in the partitioned direction. The parameter $coef$ depends on the smoother: 1 for damped Jacobi, 2 for zebra and 4 for four-color. P is the number of processors. So, the critical level is the coarsest level at which all processors can perform the smoothing operation concurrently, or in other words, the multigrid level where each processor has one local plane in the case of a damped Jacobi smoother, two planes for a zebra update and four planes in the case of a four-color update. Below the critical level, the parallel algorithm has load-balance problems that reduce its efficiency since the number of idle processors is doubled on every level below the critical one. It also complicates

its implementation, because as we go down below the critical level, we have to dynamically rearrange the communication patterns and grid distributions.

2-D and 3-D decompositions can help to reduce this problem. For example, for 3-D decompositions, the critical level is defined as the finest level L where the following condition is satisfied:

$$(\frac{N_x(L)}{coef \cdot P_x} \vee \frac{N_y(L)}{coef \cdot P_y} \vee \frac{N_z(L)}{coef \cdot P_z}) = 1 \tag{8.2}$$

where $N_x(L)$, $N_y(L)$, $N_z(L)$ are the local number of cells per side on level L in direction x, y and z respectively, and P_x, P_y and P_z are the number of processors in direction x, y and z respectively.

Without losing generality, we can assume $P_x = P_y = P_z = P^{-1/3}$, where P is the total number of processors, and $N_x(L)= N_y(L)=N_z(L) = N(L)$. Therefore, the critical level L satisfied the following condition:

$$\frac{N(L)}{coef \cdot P^{-1/3}} = 1 \tag{8.3}$$

Comparing expressions (8.1) and (8.3), it is evident that the critical level is coarser in 2-D and 3-D decompositions. In addition, for a 3-D regular application the communication requirements for a process grow in proportion to the size of the boundaries, while computations grow in proportion to the size of the entire partition. The communication to computation ratio is thus a surface area to volume ratio and so the 3-D decomposition leads to a lower inherent communication-to-computation ratio. However, 3-D decompositions require non-contiguous boundaries and, as discussed in [32, 33], the effective bandwidth is reduced due to non-unit-stride access. In fact, 2-D data partitioning is found to be a trade-off between the improvement of the message data's locality and the efficient exploitation of the underlying communication system. However, as we have explained above, 2-D decompositions are not satisfactory when a plane-wise smoother is employed.

Some alternatives, which can be applied to 1-D decompositions, have been proposed to relieve the critical-level problem. Among these, we can mention in particular:

- *Agglomeration on coarsest grids* [28]. Multigrid may be faster using only one processor on the coarsest grids (below the critical level) because the communication overhead is reduced. In our application, it increases the overall execution time since in using x-semicoarsening and a plane-wise smoother, the time spent by the sequential algorithm on the coarser grids is not negligible.

- *Parallel superconvergent multigrid* [15, 27, 14]. This approach keeps the processors busy below the critical level using multiple coarse grids. Although it slightly increases the execution time, since extra work is necessary to merge the solutions of the different grids, it may improve the convergence properties of the method. Nevertheless, in our case, we have not found any satisfactory merging operator.

- *U-Cycle method* [44]. This approach avoids idle processors fixing the number of grid levels so that the coarsest grid employed is the critical one. However, the number of iterations needed to solve the system of equations on the coarsest problem grows with system size, and the time expended on the coarsest level becomes dominant [34, 37, 38].

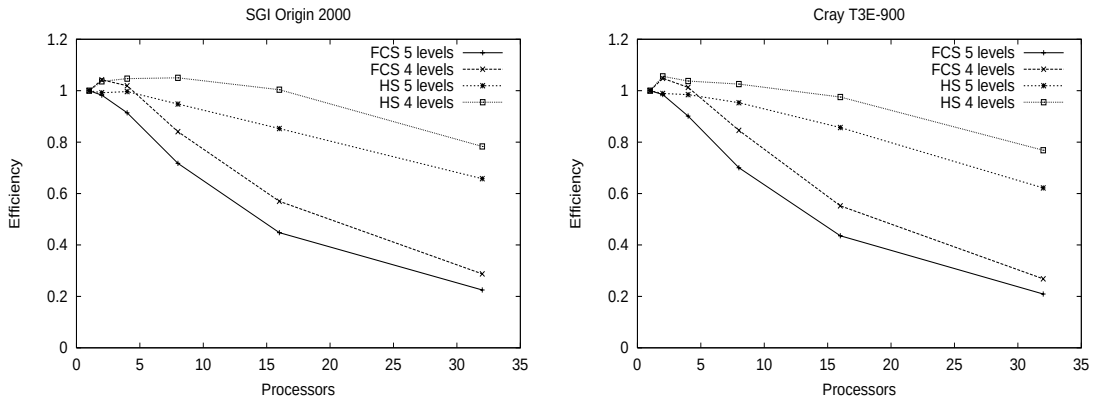

FIG. 12. *Parallel efficiency of four-color smoother (FCS) and hybrid smoother (HS) V(1,1)-cycles obtained on a SGI Origin 2000 (left-hand chart) and a Cray T3E (right-hand chart) using* 64^3 *uniform grids for up to 32 processors.*

8.4. Hybrid Smoother Strategy. We have opted to relieve the critical level problem by changing the smoother operator. As is well known, the four-color smoother (FCS) exhibits finer granularity than other common smoothers such as damped Jacobi or zebra. So, in order to improve the granularity of the solver, we have implemented a hybrid smoother (HS) that uses a four-color update in and above the critical level, a zebra smoother at the level where every processor has two planes, and a damped Jacobi method below that level. As we will show, although this smoother degrades the convergence properties of the method, it improves the execution time of the multigrid cycle. Taking both factors into account, this approach can be seen as a trade-off between the numerical and the arquitectural properties.

Figure 12 shows the efficiency of one V(1,1) multigrid cycle on a 64^3 uniform grid obtained on the T3E and O2K systems using both FCS and HS. As usual, the efficiency has been defined as $\frac{T_s}{P*T_p}$, where T_s is the execution time of the V(1,1) sequential cycle using FCS and five multigrid levels, P is the number of processors and T_p is the execution time of the parallel V(1,1) cycle.

Obviously, the lower the number of levels, the higher the efficiency, because the computation-communication ratio is higher. A similar efficiency pattern is exhibited in both systems because the inefficiency is mainly due to the load imbalance below the critical level. The overhead due to interprocessor communication is very low. Notice that the efficiency does not decrease to below 1 when the coarsest level is finer than the number of processors. Although we have included results for 32-processor simulations, it is obvious that a 64^3 problem is not big enough to obtain satisfactory efficiencies in this case. As expected, HS presents higher efficiencies because its granularity is higher below the critical level.

Despite this, as Figure 13 shows, the convergence rate per multigrid cycle strongly depends on the smoother. Consequently, as the convergence factor of the parallel algorithm may differ from that of the sequential version, in order to compare both alternatives we should use another definition of efficiency which includes both numerical (convergence) and architectural (parallel) properties. We have opted to use an efficiency definition based on the execution time needed to solve the whole problem, i.e. to reach a certain residual norm. In particular, we have chosen 10^{-12} and the corresponding efficiency will be referred to hereafter as the *realistic parallel efficiency* [25]. Figure 14 shows this realistic efficiency on both systems.

In the FCS approach, the *realistic efficiency* is similar to the efficiency of one V(1,1) cycle, since the convergence rate does not suffer any significant deterioration. This is not the case of the HS approach. However, despite the convergence deterioration caused by this technique, this is the best choice on both

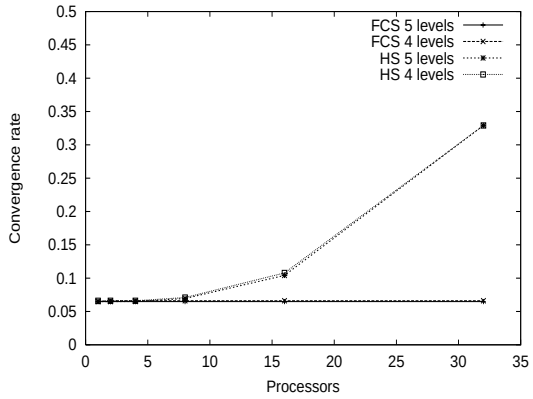

FIG. 13. *Convergence factor of four-color smoother (FCS) and hybrid smoother (HS) V(1,1)-cycles on a* 64^3 *uniform grid for different multigrid levels.*

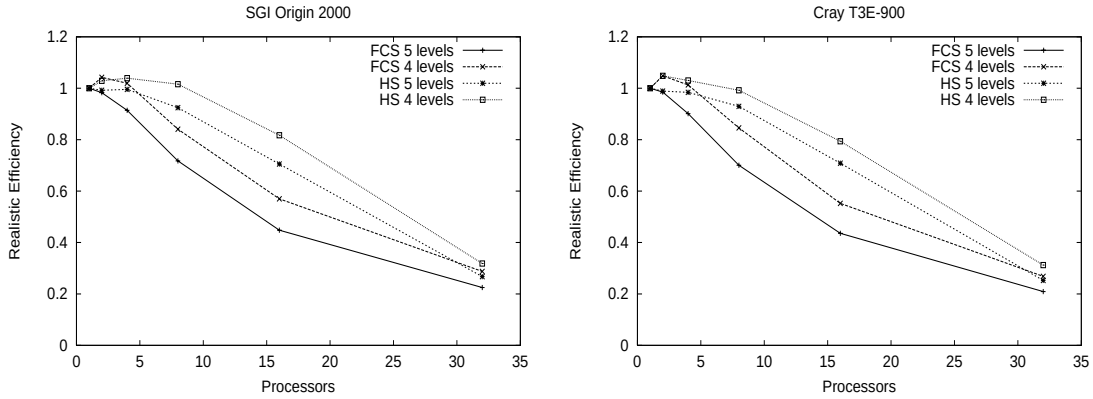

FIG. 14. *Realistic parallel efficiency of four-color smoother (FCS) and hybrid smoother (HS) V(1,1)-cycles obtained on a SGI Origin 2000 (left-hand chart) and a Cray T3E (right-hand chart) using* 64^3 *uniform grids for up to 32 processors.*

systems. Figure 15 shows the realistic efficiency on the O2K for the 128^3 problem size. Due to memory limitations, we cannot obtain results on the T3E for bigger problems. In this case, an HS with five multigrid levels, a trade-off between numerical and parallel properties, is the best choice and satisfactory efficiencies (higher than 0.8) are obtained for up to 32 processors.

9. Conclusions and Future Work. The combination of semicoarsening, a four-color plane-implicit smoother and the introduction of explicit CCI terms in the discretization of all grids yields an efficient highly parallel multigrid solver for the convection-diffusion equation with fast grid-independent convergence rates for any angle of non-alignment. This solver permits the parallel solution of a convective process that is sequential in nature.

We have opted to employ a 1-D grid partitioning in the semicoarsened direction. This solution avoids the parallel implementation of the plane solvers that has been previously reported to have a low efficiency for small problems. One-dimensional x-semicoarsening is the best choice in terms of message passing since the effective communication bandwidth on the systems under study (Cray T3E and SGI Origin 2000) presents a significant dependence on the spatial locality of the messages. Below the critical level, the parallel algorithm has load-balance problems that reduce its efficiency since the number of idle processors is doubled on every

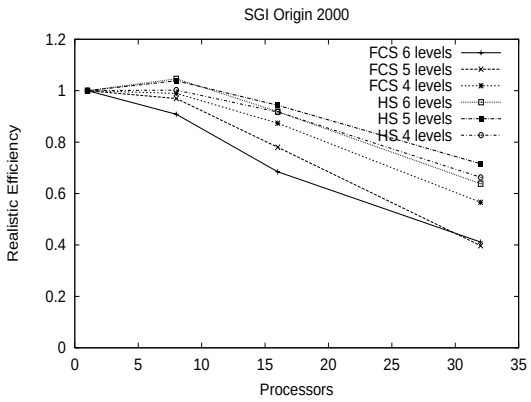

FIG. 15. *Realistic parallel efficiency of four-color smoother (FCS) and hybrid smoother (HS) V(1,1)-cycles obtained on an SGI Origin 2000 using* 128^3 *uniform grids for up to 32 processors.*

level below the critical one. It also complicates its implementation, because as we go down below the critical level, we have to dynamically rearrange the communication patterns and grid distributions.

Different alternatives have been studied in order to avoid the critical level problem: agglomeration on coarsest grids, the parallel superconvergent method and the U-cycle approach. Finally, the critical level problem is relieved by using a hybrid smoother that applies the optimal smoother on each level in terms of its convergence rate and granularity. The hybrid smoother uses a four-color update on and above the critical level, a zebra smoother at the level where every processor has two planes, and a damped Jacobi method below that level. Although this smoother degrades the convergence properties of the method, it improves the execution time of the multigrid cycle in a parallel setting. Taking both factors into account, this approach is found to be an optimal trade-off between the numerical and the arquitectural properties. Satisfactory efficiencies (higher than 0.8) are obtained for up to 32 processors on a 128^3 uniform grid.

Although not included in this report, experiments with pointwise smoothers were also performed. On finest grids, where the direction of the strongest coupling approximately coincides with the characteristic direction, a pointwise smoother can be as efficient as a plane-implicit smoother. Using pointwise smoothers on the finest (most expensive) grids considerably reduces the work-unit count of the approach. The coupling analysis provides a criterion to switch between point and plane smoothers in the multigrid cycle. Such an approach was successfully applied in [7] for 2-D vertex-centered grids and its extension for 3-D cell-centered grids will be a subject for future studies. We intend to continue the work on the effective solution of convection-dominated problems. In particular, we will apply the CCI correction technique to improve the parallel properties and convergence rate of the multigrid resolution of the Navier-Stokes equations by distributive smoothers.

10. Acknowledgments. We would like to thank Ciemat, CSC (Centro de Supercomputación Complutense), ICASE and the Computational Modeling and Simulation Branch at the NASA Langley Research Center for providing access to the parallel computers that have been used in this research.

REFERENCES

[1] E. ANDERSON, J. BROOKS, C. GRASSL, AND S. SCOTT, *Performance of the CRAY T3E multiprocessor*, in Proceedings of Supercomputing '97, ACM - IEEE, November 1997.

[2] A. Brandt, *Multigrid solvers for non-elliptic and singular-perturbation steady-state problems.* The Weizmann Institute of Science, Rehovot, Israel, December 1981, (unpublished).

[3] A. Brandt and B. Diskin, *Multigrid solvers for nonaligned sonic flows*, SIAM J. Sci. Comp., 21 (2000), pp. 473–501.

[4] A. Brandt and I. Yavneh, *On multigrid solution of high-Reynolds incompressible entering flow*, J. Comput. Phys., 101 (1992), pp. 151–164.

[5] D. E. Culler, J. P. Singh, and A. Gupta, *Parallel Computer Architecture. A Hardware/Software Approach*, Morgan Kaufmann Publishers, 1999.

[6] P. M. de Zeeuw, *Matrix-dependent prolongations and restrictions in a blackbox multigrid solver*, J. Comput. Appl. Math., 33 (1990), pp. 1–27.

[7] B. Diskin, *Solving upwind-biased discretizations II: Multigrid solver using semicoarsening*, ICASE Report 99-25, July 1999.

[8] B. Diskin and J. L. Thomas, *Half-space analysis of the defect-correction method for fromm discretization of convection.* To appear in SIAM J. Scient. Comp.

[9] ——, *Solving upwind-biased discretizations: Defect-correction iterations*, ICASE Report 99-14, March 1999.

[10] C. C. Doublas, J. Hu, U. Rüde, and C. Weib, *Cache optimizations for structured and unstructured grid multigrid*, in Proceedings of the PDCS'98 Conference, July 1998.

[11] C. C. Douglas, S. Malhotra, and M. H. Schultz, *Transpose free alternating direction smoothers for serial and parallel methods.* MGNET at http://www.mgnet.org/, 1997.

[12] D. Espadas, M. Prieto, I. M. Llorente, and F. Tirado, *Parallel resolution of alternating-line processes by means of pipelining techniques*, in Proceedings of the 7th. Euromicro Workshop on Parallel and Distributed Processing, IEEE Computer Society Press, February 1999, pp. 289–296.

[13] ——, *Solution of alternating-line processes on modern parallel computers*, in Proceedings of the 28th. annual Conference on Parallel Processing (ICPP '99), IEEE Computer Society Press, September 1999, pp. 208–215.

[14] P. Frederickson, *Recent result on pumg*, in Proceedings of the 9th Copper Mountain Conference on Multigrid Methods, 1999.

[15] P. Frederickson and O. McBryan, *Parallel superconvergent multigrid*, in Lecture Notes in Pure and Applied Mathematics, Vol 110. Marcel Dekker, New York, 1988, pp. 196–210.

[16] J. C. Agüi and J. Jimenez, *A binary tree implementation of a parallel distributed tridiagonal solver*, Parallel Comput., 21 (1995), pp. 233–241.

[17] J. E. Dendy Jr., *Black box multigrid for nonsymmetric problems*, Appl. Math. Comput., 13 (1983), pp. 261–283.

[18] J. López, O. Platas, F. Argüello, and E. L. Zapata, *Unified framework for the parallelization of divide and conquer based tridiagonal systems*, Parallel Comput., 23 (1997), pp. 667–686.

[19] Krechel, Plum, and Stüben, *Parallelization and vectorization aspects of the solution of tridiagonal linear systems*, Parallel Comput., 14 (1990), pp. 31–49.

[20] J. Laudon and D. Lenoski, *The SGI Origin: A ccNUMA highly scalable server*, in Proceedings of the International Symposium on Computer Architecture (ISCA '97), June 1997.

[21] I. M. Llorente, B. Diskin, and N. D. Melson, *Alternating plane smoothers for multiblock grids.* To appear in SIAM J. Sci. Comp.

[22] ——, *Plane-smoothers for multi-block grids: Computational aspects*, ICASE Report 99-17, May 1999.

[23] I. M. Llorente and N. D. Melson, *Behavior of plane relaxation methods as multigrid smoothers*, Electronic Transactions on Numerical Analysis, 10 (2000), pp. 92–114.

[24] I. M. Llorente and F. Tirado, *Relationships between efficiency and execution time of full multigrid methods on parallel computers*, IEEE Trans. on Parallel and Distributed Systems, 8 (1997), pp. 562–573.

[25] I. M. Llorente, F. Tirado, and L. Vázquez, *Some aspects about the scalability of scientific applications on parallel architectures*, Parallel Comput., 22 (1996), pp. 1169–1195.

[26] N. Mattor, T. J. Williams, and D. W. Hewett, *Algorithm for solving tridiagonal matrix problems in parallel*, Parallel Computing, 21 (1995), pp. 1769–1782.

[27] O. A. McBryan, P. O. Frederickson, J. Linden, A. Schüller, K. Solchenbach, K. Stüben, C. Thole, and U. Trottenberg, *Multigrid methods on parallel computers-a survey of recent developments*, Impact of Computing in Science and Engineering, 3 (1991), pp. 1–75.

[28] D. Moulton and J. Dendy, *Mpi-based black box multigrid for workstation clusters*, in Proceedings of the 9th Copper Mountain Conference on Multigrid Methods, 1999.

[29] C. W. Oosterlee, F. J. Gaspar, T. Washio, and R. Wienands, *Multigrid line smoothers for higher order upwind discretizations of convection-dominated problems*, J. Comput. Phys., 139 (1998), pp. 274–307.

[30] A. Povitsky, *Parallel directionally split solver based on reformulation of pipelined thomas algorithm*, ICASE Report 98-45, October 1998.

[31] ——, *Parallelization of the pipelined thomas algorithm*, ICASE Report 98-48, December 1998.

[32] M. Prieto, D. Espadas, I. M. Llorente, and F. Tirado, *Message passing evaluation and analysis on Cray T3E and SGI Origin 2000 systems*, in Proceeedings of the 4th. Euro-Par Conference (Euro-Par '99), Lecture Notes in Computer Science, Vol. 1685. Springer. Berlin, August 1999, pp. 173–182.

[33] M. Prieto, I. M. Llorente, and F. Tirado, *Partitioning regular domains on modern parallel computers*, in Proceedings of the 3rd. International Meeting on Vector and Parallel Processing (VECPAR '98), Lecture Notes in Computer Science, Vol. 1573. Springer. Berlin, 1998, pp. 411–424.

[34] M. Prieto, R. Santiago, I. M. Llorente, and F.Tirado, *A parallel robust multigrid algorithms based on semicoarsening*, in Proceedings of the 6th European PVM/MPI Users' Group Meeting, Lecture Notes in Computer Science, Vol. 1697. Springer. Berlin, September 1999, pp. 307–316.

[35] D. Quinlan, F. Bassetti, and D. Keyes, *Temporal locality optimizations for stencil operations within parallel object-oriented scientific frameworks on cache-based architectures*, in Proceedings of the PDCS'98 Conference, July 1998.

[36] U. Rüde, *Iterative algorithms on high performance architectures*, in Proceedings of the Europar'97 Conference, 1997, pp. 57–71.

[37] R. Santiago, M. Prieto, I. M. Llorente, and F.Tirado, *A robust multigrid solver on parallel computers*, Springer-Verlag in the Lecture Notes in Computer Science and Engineering series. To appear in Post-conference book of the European Multigrid Meeting 1999.

[38] ——, *Robust multigrid algorithms for 3-D elliptic equations on structured grids*, in Proceedings of the 8th. Euromicro Workshop on Parallel and Distributed Processing, IEEE Computer Society Press, January 2000, pp. 48–55.

[39] SGI, *Document number 007-3430-002, Origin 2000 and Onyx2 Performance Tuning and Optimization Guide*, available at http://techpubs.sgi.com, 1999.

[40] B. Smith, P. Bjorstad, and W. Gropp, *Domain Decomposition, Parallel Multilevel Methods for*

Elliptic Partial Differential Equations, Cambridge University Press, 1996.

[41] J. L. Thomas, B. Diskin, and A. Brandt, *Distributed relaxation multigrid and defect correction applied to the compressible Navier-Stokes equations*, AIAA Paper 99-3334, 14th Computational Fluid Dynamics Conference, Norfolk, VA, July 1999.

[42] ——, *Textbook Multigrid Efficiency for the Incompressible Navier-Stokes Equations: High Reynolds Number Wakes and Boundary Layers*, ICASE Report 99-51, December 1999.

[43] H. H. Wang, *A parallel method for tridiagonal equations*, ACM Trans. on Math. Software, 7 (1981), pp. 170–183.

[44] D. Xie and L. R. Scott, *The parallel u-cycle multigrid method*, in Proceedings of the 8th Copper Mountain Conference on Multigrid Methods, 1996.

[45] N. Yanenko and Y.I.Shokin, *On the correctness of first differential approximations of difference schemes*, Dokl. Akad. Nauk SSSR, 182 (1968), pp. 776–778.

[46] I. Yavneh, *Coarse-grid correction for non-elliptic and singular perturbation problems*, SIAM J. Sci. Comput., 19 (1998), pp. 1682–1699.

[47] S. Zeng, C. Vink, and P. Wesseling, *Multigrid solution of the incompressible Navier-Stokes equations in general coordinates*, SIAM J. of Num. Anal., 31 (1994), pp. 1764–1784.